Dokumentacja Projektu I

Przedmiot: Programowanie obiektowe (w języku C#)

Autorka: Katarzyna Bąk

Numer albumu: 48678

Semestr II

Spis treści

[1. Informacje o projekcie 3](#_Toc4334957)

[2. Badanie zbieżności szeregu 3](#_Toc4334958)

[3. Wygląd formularza i lokalizacja kontrolek 4](#_Toc4334959)

[4. Zmienne, klasy i metody 4](#_Toc4334960)

[1. Zmienne i stałe 4](#_Toc4334961)

[2. Klasa 5](#_Toc4334962)

[3. Metody 5](#_Toc4334963)

[5. Rysowanie w programie formularzowym 6](#_Toc4334964)

[6. Animacja komputerowa – narzędzia w języku C# i ich zastosowanie w projekcie 6](#_Toc4334965)

[7. Testowanie wybranych funkcjonalności 6](#_Toc4334966)

# Informacje o projekcie

Program, którym wykonałam podczas realizacji Projektu I na przedmiot Programowanie obiektowe jest to program formularzowy napisany w języku C#. Polega on na animacji ruchu kulki po szeregu oraz możliwości zmiany atrybutów elementów znajdujących się na formularzu, takich jak:

* zmiana koloru kulek, linii szeregu oraz linii układu współrzędnych
* zmiana prędkości ruchu kulki
* zatrzymanie i ponowne uruchomienie kulki
* zmiana grubości linii szeregu
* zmiana promienia kulki
* zmiana stylu linii wykresu oraz osi X i Y

Szereg, po którym porusza się kulka ma następujący wzór:

Do narysowania linii szeregu oraz wykresu wykorzystałam fakt, że C# jest językiem obiektowym i umożliwia on narysowanie wykresu w programach formularzowych bez użycia kontrolki *Chart.* Właściwości języka C# oraz dołączona do projektu biblioteka *Drawing2D* pozwalają rysować różne obiekty graficzne na formularzu, takie jak: linie proste, linie krzywe, kształty.

W projekcie użyłam klas statycznych, którymi posłużyłam się do przeliczenia współrzędnych w celu narysowania układu współrzędnych w konkretnym przedziale oraz do wyliczenia odpowiednich wartości szeregu. Klasy statyczne umożliwiają nam tworzenie metod oraz zmiennych statycznych, które mogą być wykorzystywane uniwersalnie w całym projekcie.

Jednak aby właściwie wyrysować linię szeregu, potrzebne jest obliczenie jego zbieżności, ponieważ możemy rysować szereg tylko w przedziale jego zbieżności (wyliczenia znajdują się w następnym punkcie).

# Badanie zbieżności szeregu

Powyższy szereg jest szeregiem potęgowym. Dla szeregów potęgowych, do przedziału zbieżności bierzemy pod uwagę tylko wyrażenie, w którym występuje: .

Szereg jest zbieżny dla , co dla powyższego szeregu wyznacza się w następujący sposób:

Z tego wynika, że szereg jest zbieżny dla przedziału .

# Wygląd formularza i lokalizacja kontrolek

Lokalizacja formularza i jego wymiary są wyznaczane przez funkcje bibliotek języka C#, które automatycznie pobierają rozdzielczość ekranu komputera i na jej podstawie wyznaczają rozmiar okna formularza. Gwarantuje to pełną uniwersalność rozmiarów formularza niezależnie od rozdzielczości ekranu używanego przez użytkownika komputera.

Ustawienie kontrolek znajdujących się na formularzu ustalana jest ręcznie w kodzie programu. Jest to spowodowane tym, aby ich lokalizacja była uniwersalna, nie zmieniała się w momencie, gdy użytkownik zmieni rozmiar formularza.

# Zmienne, klasy i metody

Przedstawiony projekt opiera się na odtworzeniu na płaszczyźnie układu współrzędnych szeregu potęgowego. Jednak trzeba pamiętać, że w programowaniu układ współrzędnych jest odwróconą pierwszą ćwiartką rzeczywistego układu współrzędnych. Narysowanie go w programie wymaga więc przeliczenia wielu jednostek w celu odtworzenia docelowego fragmentu układu. Jest to przygotowanie do ostatecznego wyrysowania układu, linii szeregu oraz kulki, która będzie się po niej poruszać na formularzu. Tak jak w poprzednim punkcie zostało wyjaśnione, szereg został wyrysowany dla . Wartości są natomiast z przedziału od ok. -0,45 do ok. 3,8.

W związku z tymi wartościami, interesują nas jedynie początkowe fragmenty I oraz IV ćwiartki rzeczywistego układu. Dla przeskalowania wartości użyłam następujących zmiennych i metod, które zaprezentuję w tabelce poniżej.

## Zmienne i stałe

|  |  |  |
| --- | --- | --- |
| Nazwa | Typ | Zastosowanie |
| kbMargines | const int | Stała, która odpowiada za rozmiar marginesu |
| kbPrzyrost | const int | Stała, która odpowiada za przyrost zmiennej x |
| kbPromienKulki | int | Zmienna wyznaczająca promień kulki |
| kbXMax, kbYMax | int | Maksymalne wartości wymiarów okna formularza (pomniejszone o margines z prawej strony)  wyznaczające rozmiar powierzchni graficznej |
| kbXs, kbYs | int | Połowa wymiarów okna formularza |
| kbWspolczynnikSkaliX, kbWspolczynnikSkaliY | float | Współczynniki pomagające przeskalować osie do poprawnego wyświetlania wykresu |
| kbIndeksPunktow | int | Indeks punktów na szeregu |
| kbIndexPolozeniaKulek | int | Indeks położenia kulek na szeregu |
| kbJednostki | int | Obecne położenie x, które wyrażone jest w ilości pikseli od początku układu współrzędnych |
| kbn | int | Zmienna pomocnicza dla wyznaczenia wartości szeregu |

## Klasa

|  |  |  |
| --- | --- | --- |
| kbPrzeliczanieWspolrzednych | static class | Klasa statyczna, w której zdefiniowane są metody używane w celu przeliczania współrzędnych |

## Metody

|  |  |  |
| --- | --- | --- |
| kbSrodek | int | Metoda służąca do wyznaczenia środka układu współrzędnych |
| kbWspolczynnikSkaliX, kbWspolczynnikSkaliY | float | Metody służące do wyznaczania współczynników skalowania dla odpowiednich osi |
| kbWartoscSzeregu | double | Metoda służąca do wyznaczenia wartości szeregu |
| kbWartoscY | double | Metoda służąca do wyskalowania wartości szeregu względem osi OY |
| kbIloscPunktow | int | Metoda wyznaczająca ilość punktów szeregu zależnie od przyrostu |
| kbPrzyrostPikseli | int | Metoda wyznaczająca, co ile pikseli zmienia się wartość x |

Wyjaśnienia:

* *const* – powoduje, że wartość jest stała przez cały czas działania programu
* *int* – zmienna całkowita
* *float* – zmienna zmiennoprzecinkowa
* *double* – zmienna zmiennoprzecinkowa o większym zakresie niż *float*. Jest wykorzystywana przy liczeniu pierwiastków
* *static* *class* – rodzaj klasy służący do przechowywania zmiennych i metod, które mogą być wykorzystywane w całym projekcie. Nie można tworzyć jej obiektów

# Rysowanie w programie formularzowym

Po przeliczeniu współrzędnych można przejść do rysowania na formularzu. Jest to możliwe przez dodanie biblioteki *Drawing.Drawing2D,* która udostępnia nam wiele przydatnych klas. Klasy te umożliwiają tworzenie nowych obiektów oraz nadawanie im atrybutów. Oto klasy, którymi posłużyłam się w projekcie wraz z opisem i celem ich użycia:

* *Pen* – utworzenie obiektów piór: kbPioroXY, kbPioroSzereg do wyrysowania osi i wykresu szeregu
* *Graphics* – utworzenie obiektu kbRysownica w celu utworzenia powierzchni graficznej
* *SolidBrush* – utworzenie obiektów kbPedzelKulka do wyrysowania kulki na formularzu oraz kbPedzelOpisuOsi do opisania osi układu współrzędnych
* *Point* - utworzenie obiektów dla rozmieszczenia kontrolek na formularzu, opisu osi, lokalizacji formularza na ekranie. Utworzona jest również tablica punktów szeregu.
* *PointF* - utworzenie obiektów dla opisu osi X i Y
* *AdjustableArrowCap* - utworzenie obiektu kbNowyGrotStrzalki do zdefiniowania własnego grotu strzałki dla osi

Ostateczne wyrysowanie wszystkich obiektów zdefiniowane jest w obsłudze zdarzenia *Paint* formularza. Służą do tego metody *DrawLine* – do wyrysowania osi układu współrzędnych, *DrawCurve* – do wyrysowania linii szeregu, *FillElipse* – do rysowania kulek oraz *DrawString* do opisu osi układu współrzędnych.

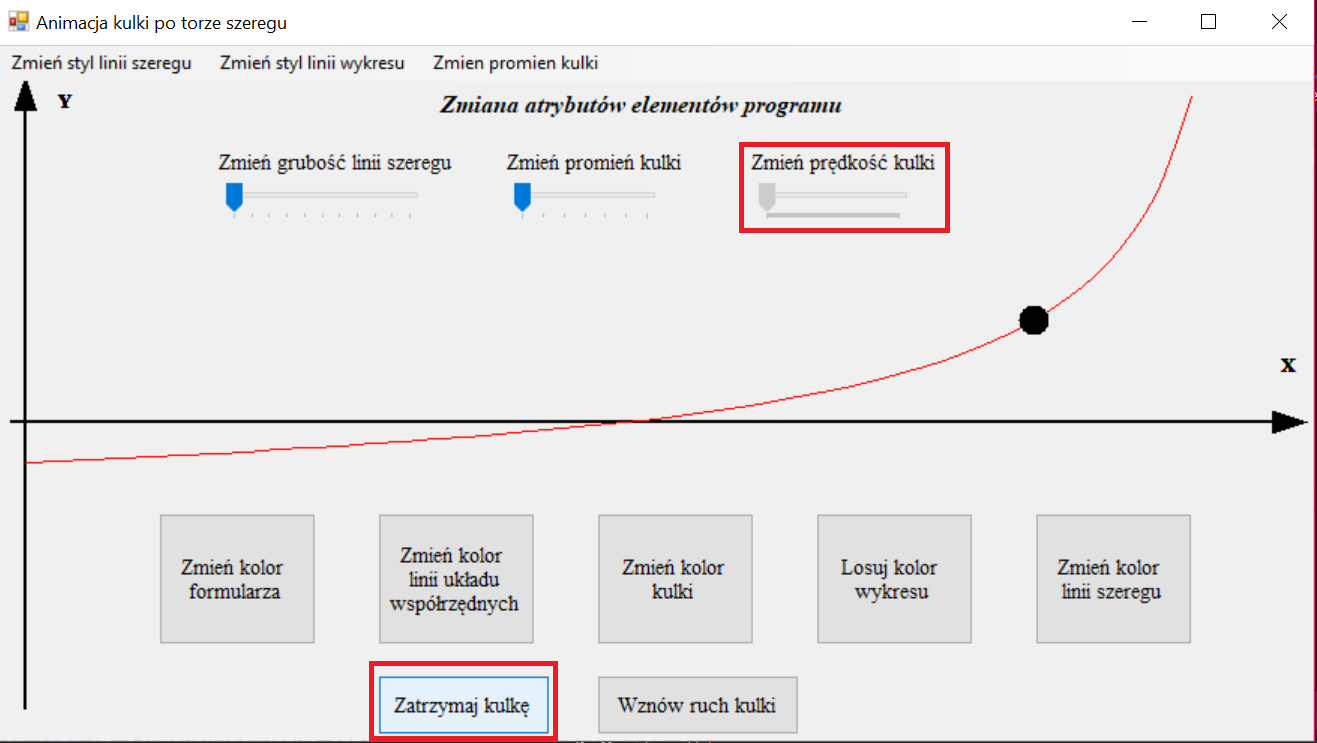
# Animacja komputerowa – narzędzia w języku C# i ich zastosowanie w projekcie

Animacja komputerowa polega na wyświetleniu kolejnych elementów po sobie w określonych odstępach czasu tak, aby efekt końcowy sprawiał wrażenie ruchu. Związana jest ona ze zmianą położenia jakiegoś konkretnego obiektu w czasie.

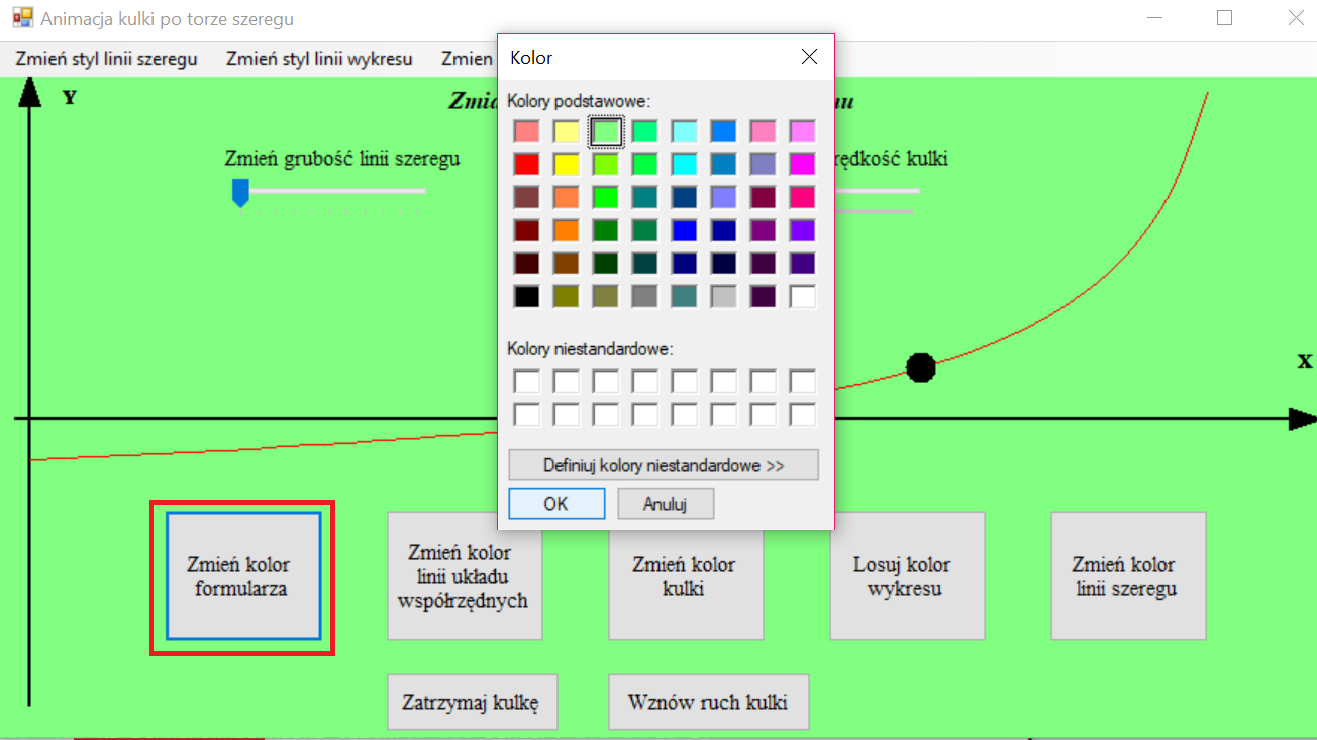
W moim projekcie obiektem, który zmienia swoje położenie jest kulka. Wykreślona jest ona na linii szeregu, gdzie pojawia się w kolejnych punktach. Narzędziem, które zostało przeze mnie wykorzystane jest kontrolka *timer*. Umożliwia ona odświeżenie formularza po wywołaniu funkcji *Refresh*. Funkcja zostaje wykonana co tzw. „tik” zegara co oznacza, że po odmierzeniu wartości interwału zegara(liczonej w milisekundach) formularz odświeży się i kulka zostanie przesunięta do następnej lokalizacji. Im mniejsza wartość interwału, tym szybciej kulka przechodzi do nowej lokalizacji, co sprawia wrażenie bardziej płynnego ruchu.

# Testowanie wybranych funkcjonalności

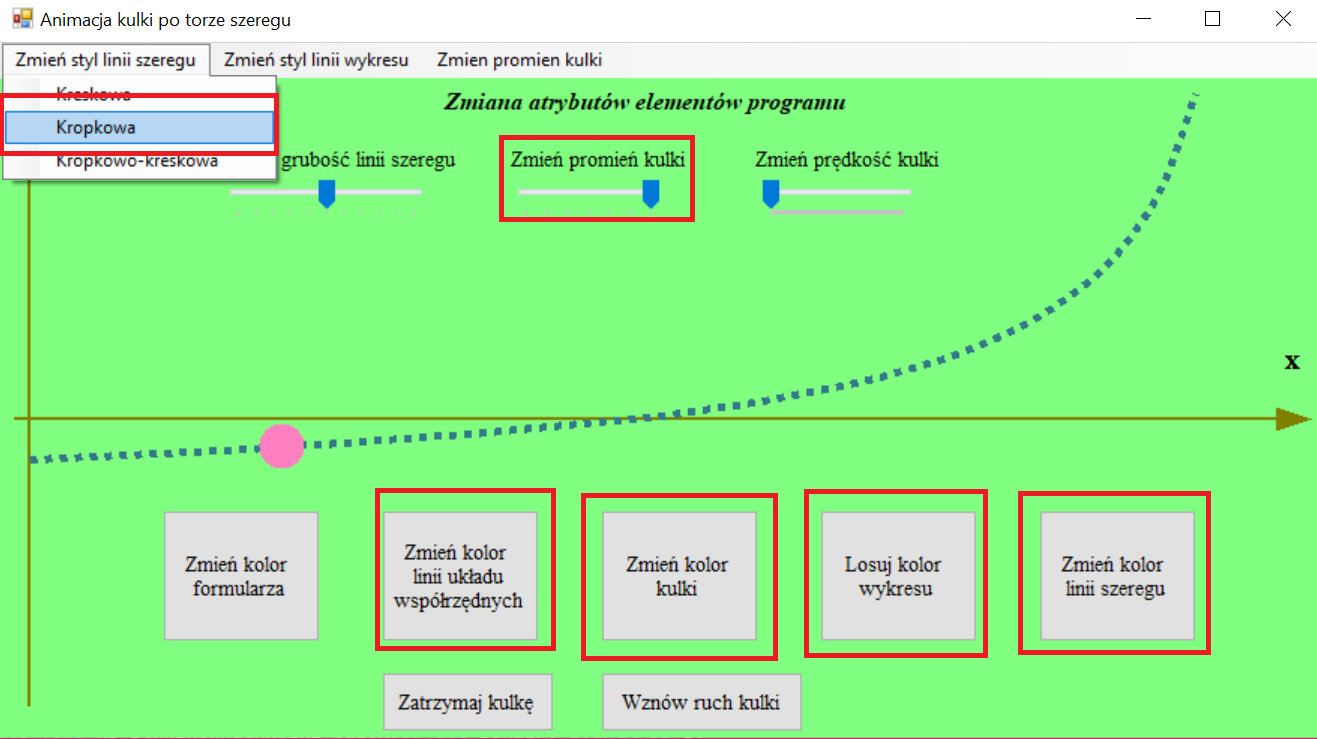
Testowanie funkcjonalności opiera się na sprawdzeniu, czy wybrana kontrolka działa poprawnie. Oto screeny testów wraz z komentarzami:



Naciśnięcie przycisku *Zatrzymaj kulkę* powoduje zatrzymanie poruszającego się po linii obiektu. Jednocześnie suwak do zmiany prędkości kulki zostaje zablokowany.

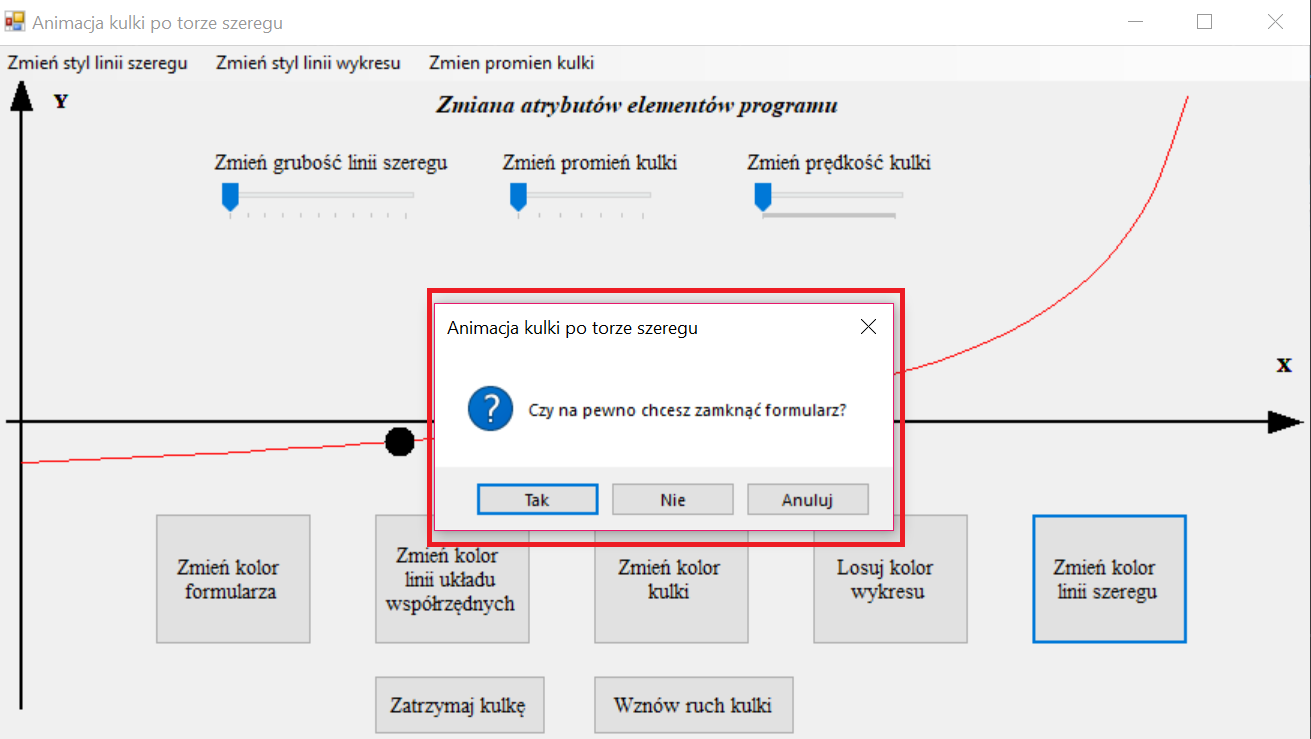


Przycisk *Zmień kolor formularza* powoduje pojawienie się okna dialogowego do zmiany koloru tła. Na screenie widoczne jest okno oraz zaznaczony przycisk. Można też zauważyć, że w oknie dialogowym zaznaczony kolor automatycznie zmienia się do wybranego koloru formularza.



Następuje zmiana różnych atrybutów:

* zmiana linii na kropkową – kontrolka *MenuStrip*
* zmiana promienia kulki za pomocą suwaka
* zmiana kolorów kulki, linii układu współrzędnych
* zmianę koloru szeregu umożliwiają dwa przyciski: *Losuj kolor wykresu* oraz *Zmień kolor szeregu*. Pierwsza kontrolka losuje randomowe liczby, które są z zakresu barw RGB, a druga otwiera okno dialogowe



Przy próbie zamknięcia formularza pojawia się okienko z zapytaniem o słuszność podjętej decyzji – Czy użytkownik na pewno chce zamknąć formularz.